VŠB – Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zásady designu uživatelských rozhraní

Principles of User Interfaces Design

2013

David Kotlář

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání bakalářské práce

Student: **David Kotlář**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R059 Mobilní technologie

Téma: Zásady designu uživatelských rozhraní
Principles of User Interfaces Design

Zásady pro vypracování:

Dobře navržené uživatelské rozhraní umožňuje uživateli pracovat s aplikací účelně a pohodlně. Pokud navíc dodržuje předepsaná pravidla, uživatel si vytváří správné stereotypy při práci se systémem.
Cílem této bakalářské práce je shrnutí všeobecně platných pravidel pro tvorbu grafických uživatelských rozhraní a zhodnocení uživatelského rozhraní vybrané aplikace včetně realizace jejího grafického rozhraní pro prostředí Metro, používané v systémech Windows Phone a Windows 8.
Práce bude obsahovat následující body:

1. Popište pravidla návrhu uživatelských rozhraní.
2. Uveďte, a na příkladech popište, nejčastější chyby při návrhu uživatelských rozhraní.
3. Vyberte si existující, prakticky využívanou aplikaci a zhodnoťte její uživatelské rozhraní.
4. Navrhněte a realizujte rozhraní vybrané aplikace pro prostředí Metro.
5. Zhodnoťte dosažené výsledky.

Seznam doporučené odborné literatury:

[1] Ben Shneiderman, Catherine Plaisant: *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison-Wesley, 2004
[2] Alan Dix, Gregory D. Abowd, Russell Beale: *Human-Computer Interaction 3rd Ed.*, Prentice Hall, 2003
[3] John M. Carroll: *HCI Models, Theories, and Frameworks*, Morgan Kaufmann, 2003

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Michael Holuša**

Datum zadání: 16.11.2012
Datum odevzdání: 07.05.2013

prof. RNDr. Vladimír Vašinek, CSc.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne: *2.5.2013*

.....
podpis studenta

Poděkování

Rád bych poděkoval *Michaelu Holušovi* za odbornou pomoc a konzultaci při vytváření této bakalářské práce.

Abstrakt

Základem každé aplikace je dobře navržené uživatelské rozhraní, které umožňuje uživateli pracovat účelně a pohodlně. Pokud navíc dodržuje předepsaná pravidla, uživatel si vytváří správné stereotypy při práci se systémem, snadněji a rychleji se s aplikací naučí pracovat.

Hlavními body této bakalářské práce je shrnutí všeobecně platných pravidel a zásad pro tvorbu grafických uživatelských rozhraní. Vyhledávání nejčastějších chyb, které se mohou vyskytovat v uživatelském rozhraní. Zhodnocení uživatelského rozhraní vybraného okruhu textových aplikací. Hlavním bodem práce je realizace grafického rozhraní pro nové prostředí Windows Store, používané v systémech Windows Phone a Windows 8. Poznatky z veškerých bodů při psaní této práce jsem využil k realizaci grafického uživatelského rozhraní textové aplikace WSPad pro prostředí Windows Store. Návrh uživatelského rozhraní je realizován v jazyku Xaml. Funkční část pak jazyk C#.

Klíčová slova

GUI, Windows Store, uživatelské rozhraní, textový editor, animace

Abstract

The basis of each application is well designed user interface that allows the user to work efficiently and comfortably. In addition, if adherence to the prescribed rules, the user creates the correct stereotypes when working with the system easier and faster to learn to work with the application.

The main point of this thesis is a summary of generally accepted rules and principles for creating graphical user interfaces. Search the most common errors that may appear in the user interface. Evaluation of user interface text of a selected group of applications. The main point of this work is to implement a graphical interface for the new Windows Store, used in Windows Phone and Windows 8. Knowledges from all points in writing this work I used to implement a graphical user interface for text applications WSPad Windows Store. User interface design is implemented in XAML. The functional part is implemented in language C #.

Key words

GUI, Windows Store, user interface, text editor, animation

Seznam použitých zkratek

Zkratka	Anglický význam	Český význam
GUI	Graphical User Interface	Grafické uživatelské rozhraní
WinRT	Windows Runtime	Architektura aplikací
XAML	Existencible Application Markup Language	Značkovací jazyk

Seznam použitých termínů

Termín	Význam termínu
MessageDialog	Vyskakovací okno, obvykle nese nějakou informaci o chodu aplikace
Separátor	Oddělovač mezi bloky nebo tlačítky
Undo	Tlačítko, které vrací na poslední provedenou operaci
Redo	Tlačítko, které odvolá vrácení poslední změny
StackPanel	Prvek, který uspořádá vložené komponenty do jednoho řádku
PopUp	Vyskakovací prvek aplikace
Transition	Kolekce animací prostředí Windows Store
Button	Tlačítko, pro běžné použití
Grid	Rozmísťovací prvek

Obsah

1 Úvod

Grafické uživatelské rozhraní výrazně ulehčuje práci uživateli. Je důležité, aby při tvorbě uživatelského rozhraní byly dodrženy nějaké základní zásady. K tomu slouží pravidla uživatelského rozhraní, zejména osm základních pravidel pro tvorbu dobrého GUI. V dnešní době, kdy neplatí používání příkazové řádky, jako jediného ovládacího nástroje, se veškeré problémy s uživatelským rozhraním vyskytují pouze ojediněle. U jednotlivých licencovaných či volně šiřitelných textových editorů či procesorů tomu není jinak. Může se však stát, že dojde k setkání s chybami, které zpomalují naši práci s danou aplikací právě vlivem špatného návrhu.

Windows 8 od společnosti Microsoft teprve nedávno proklouzl do obchodů jakožto první operační systém určený jak pro klasické PC, tak i pro přístroje s dotykovými displeji. Windows 8 přinesl řadu novinek oproti svému předchůdci Windows 7. Nedílnou součástí tohoto nového operačního systému je bezpochyby Windows Store, které dostalo za úkol nahradit nabídku Start. Váže na sebe i aplikace, určené přesně pro toto prostředí.

Hlavním bodem práce je však návrh a realizace uživatelského rozhraní textového procesoru pro nové prostředí Windows Store, který nese název WSPad. Jedná se o aplikaci, která obsahuje základní funkčnost pro oblast textových editorů a procesorů. Aplikaci jsem programoval v jazyku Xaml a C#.

2 Popis pravidel návrhu uživatelského rozhraní

Při tvorbě uživatelského rozhraní je nutné dodržovat jistá základní pravidla. Při pozdějším vývoji tak lze předcházet určitým chybám, které mohou ovlivnit náš konečný výsledek. Tyto pravidla rozdělujeme na osm základních pravidel, kterým se říká tzv. Osm zlatých pravidel [1]. Nesmíme však zapomenout na to, že tato pravidla nelze brát úplně doslovně. Je potřeba je v jistých situacích rozšířit, zpřísnit nebo upravit. Pravidla se však vždy snažíme dodržet, pokud není dostatek důvodu k tomu, abychom je změnili a vytvořili tak lepší uživatelské rozhraní.

2.1 Osm pravidel pro tvorbu dobrého GUI

2.1.1 Konzistence

Uživatelské rozhraní musí být konzistentní. Pokud není, zhoršuje schopnost uživatele ovládat danou aplikaci, pomaleji se s ní učí pracovat a snižuje se pochopitelně i celková spokojenost uživatelů. Aplikace se pak stává více méně chaotickou se všemi důsledky z toho vyplývajícími. Proto je důležité dodržet toto pravidlo a usilovat o konzistenci. Používat konzistentní terminologii, dodržovat pravidla pro tvorbu rozhraní daného prostředí, směřovat k tvorbě stereotypů - stejné věci se dělají stejně, podobné věci se dělají podobně. Je důležité sjednotit ovládání aplikace tak, aby v každém okně nebylo úplně jiné.

2.1.2 Informativní zpětná vazba

Zpětnou vazba je důležitá proto, že uživatel potřebuje být přiměřeně informován o výsledcích práce s aplikací. Zda daná akce proběhla úspěšně nebo neúspěšně. Pokud akce proběhne neúspěšně je důležitě se ptát proč k tomu došlo. Je nutné také zvážit formu zpětné vazby tak, aby aplikace uživatele zbytečně neobtěžovala. Informativní zpětnou vazbu dělíme na dva způsoby. Silnou a slabou zpětnou vazbu. Silná zpětná vazba je taková, kdy uživatel musí explicitně dát najevo, že vzal zprávu na vědomí nebo na ni musí přímo reagovat některou z nabízených možností. Slabá zpětná vazba je taková, kde uživatel nemusí potvrzovat, že se s informací seznámil.

2.1.3 Prevence chyb a řešení následných situací

Důležité je předcházet chybám. Pomocí uživatelského rozhraní je důležité minimalizovat případné chyby uživatele při používání aplikace. V případě, že však dojde k chybě, třeba z toho důvodu, že jí nelze předejít, je nutné uživatele o dané chybě informovat. Toto lze realizovat například pomocí chybové hlášky tzv. *MessageDialog* okna obr. (2.1), kde by mělo být srozumitelně a přehledně uvedeno o jakou chybu se jedná a možné doporučení její nápravy. Je důležitě však s uživatelem komunikovat "jeho" jazykem. Technické a jiné hlášky běžnému uživateli nijak nepomohou, jsou pro něj tedy bezcenné a mohou uživatele i nějakým způsobem frustrovat.

Obrázek 2.1: Chybový Message Dialog

2.1.4 Nabídnout "Undo" a "Redo"

Uživatel ocení, pokud bude aplikace tolerantní k jeho chybám a umožní mu tak návrat zpět. Je vhodné, aby se uživatel mohl vrátit všude tam, kde je možné vrátit akci zpět, s možností ji opětovně provést. Pokud bude tato akce vyvolána, musí být aplikace schopna se vrátit nebo jí alespoň zastavit. Je proto důležité poskytnou funkci zpět/znovu (Undo/Redo). Tyto funkce jsou podstatným rozdílem oproti znakově orientovaným rozhraním jako je například příkazová řádka, kde nelze použít příkaz vzít akci zpět.

2.1.5 Plná kontrola nad produktem pro všechny uživatele

Je nutné respektovat širokou skupinu uživatelů. Autor aplikace si musí ujasnit pro jakou, či jaké skupiny uživatelů mohou aplikaci používat (začátečníci, zkušenější uživatelé, děti či postižení lidé..). Na základě rozdělených skupin uživatelů je nutné tvořit rozhraní aplikace. Podle různých skupin je důležité určit styl práce aplikace, aby existovaly různé alternativy při ovládání aplikace.

2.1.6 Připravit produkt i pro zkušené uživatele

Vytvářet předvídatelné uživatelské rozhraní je vhodné pro zkušenější uživatele. Uživatel musí být řídicím prvkem rozhraní aplikace. Musí být iniciátorem, nesmí dojit k situaci, kdy uživatel bude plnit "rozkazy" aplikace. Jakákoliv nepředvídatelnost chování aplikace může způsobit to, že uživatel ztratí nadvládu nad produktem a stane se tak její pomyslnou obětí. Pokročilý uživatel preferuje obvykle snadnější uživatelské rozhraní. Komplikovanější uživatelské rozhraní tedy uživatele spíše zdržuje. Uživatel také uvítá různé zkratky, makra a jiné. Je nutné se držet toho, že zkušenější uživatel chce mít aplikaci a její GUI přečtené.

2.1.7 Organizace akci do uzavřených celků

Navigace aplikace má z hlediska uživatelského rozhraní také svou důležitost. Řada uživatelských úloh se sestavuje z posloupnosti více po sobě jdoucích akcí. Akce uživatelského rozhraní je nutné rozdělit na jednoduché, logicky rozčleněné kroky, které respektuji jednotlivé pracovní postupy aplikace. Komplikovanější akce s větším počtem kroků je vhodné rozdělit na menší celky, které mají jasný začátek a jasný konec. Po vykonání jednotlivého celku je doporučeno použít zpětnou vazbu.

2.1.8 Nepřetěžovat krátkodobou paměť uživatele a jeho vizuální systém

Uživatelské rozhraní je vhodné přizpůsobit tak, aby nebyla přetěžována krátkodobá a vizuální paměť uživatele. Uživatel není povinen pamatovat si doslova uživatelské rozhraní aplikace. Naopak je důležité, aby uživatelské rozhraní dávalo uživateli přehled o aplikaci, bez jakékoliv nutnosti si jednotlivé věci pamatovat. Toto pravidlo lze přirovnat k řízení automobilu. Řidič, představující uživatele potřebuje vědět co se děje před ním, vedle něj i za ním. Je nutné se vyvarovat nadměrnému počtu položek v menu, chaosu na obrazovce. Také je důležité preferovat danou strukturu obrazovky.

2.2 Vizuální uspořádání prvku pro vytvoření dobrého GUI

Vizuální uspořádání je určitě neméně důležité, jak již výše zmíněných osm zlatých pravidel. Je důležité, aby autor aplikace dokázal vytvořit srozumitelné a přehledné uživatelské rozhraní pro daného uživatele. Vizuální prvky aplikace, jako jsou konkrétní rozměry oken, použití vhodné barevné kombinace či vzdálenosti jednotlivých prvků aplikace, jako jsou různé tlačítka a jiné komponenty.

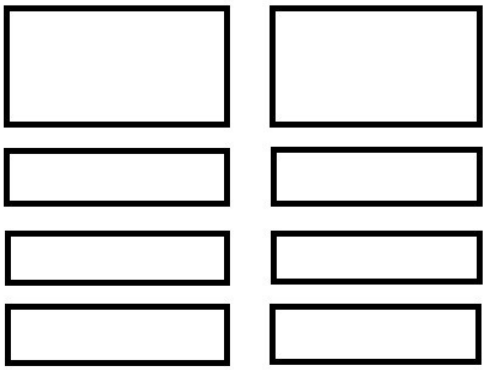
Při uspořádaní nebo vkládání prvků je důležité si uvědomit pár základních věcí:

- Minimalizovat dráhu očí, tzn. Uspořádat prvky od shora dolů, či zleva doprava.
- Uživatel vždy prohlíží obsah obrazovky z levé strany ve směru na pravou stranu.
- Rozmístění a posloupnost všech ovládacích prvků musí respektovat tok informací.

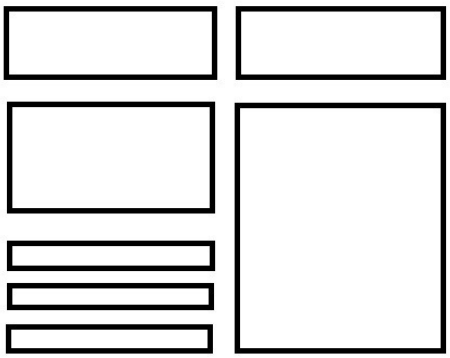
Z hlediska pohledu člověka je důležité vnímání jednoduchosti a účelnosti aplikace. Proto je dobré držet se zásad pro vizuální organizaci komponentů dané aplikace:

- **Vyváženost**

Aby aplikace byla pro uživatele přehledná je nutné dodržovat jak vertikální, tak horizontální rozměry komponent požívaných při tvorbě aplikace.



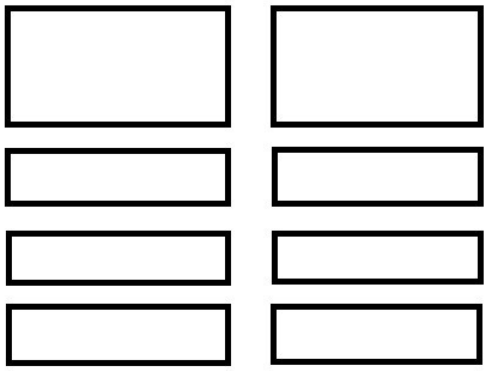
Obrázek 2.2: Vyváženost



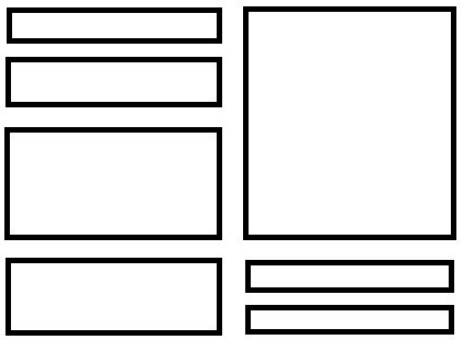
Obrázek 2.3: Nevyváženost

- **Souměrnost**

Členění skupin ovládacích prvků má být stejné, nebo podobné na levé i pravé straně okna. Nesouměrné (asymetrické) uspořádání působí rozměrnějším dojmem, než souměrné uspořádání.



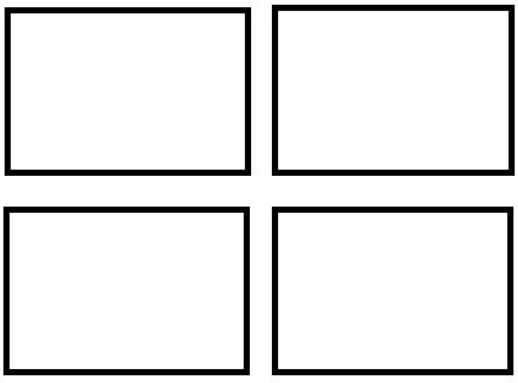
Obrázek 2.4: Souměrnost



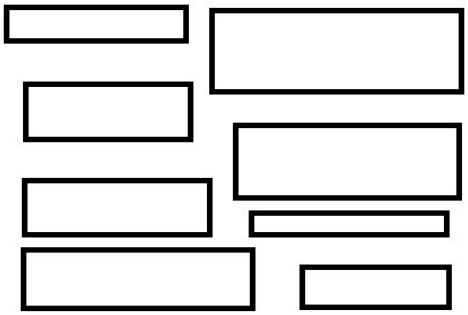
Obrázek 2.5: Nesouměrnost

- **Pravidelnost**

Rozměry stejných ovládacích prvků, jejich tvar, barva, vzdálenosti mezi nimi by měly být všude tam, kde je to vhodné jednotné. Konkrétní parametry by měly být definovány pro každý jazyk.



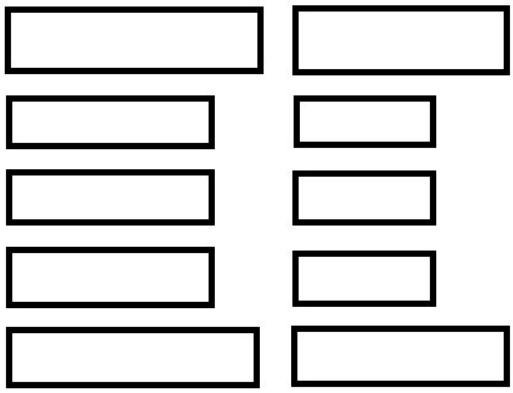
Obrázek 2.5: Pravidelnost



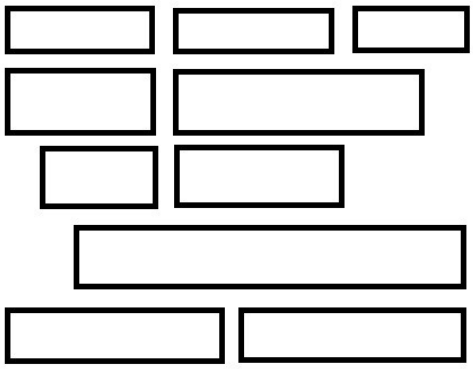
Obrázek 2.6: Nepravidelnost

- **Předvídatelnost**

Styl a logika uspořádání prvků v oknech a jejich ovládání má být totožný napříč všemi okny a v souladu s dokumentací.



Obrázek 2.7: Předvídatelnost



Obrázek 2.8: Nepředvídatelnost

- **Účelnost**

Vždy se řídíme pravidlem „*forma sleduje" funkci*. Používejte jednotný a jednoduchý styl, střídmě nakládejte s barvami, tvary a podobně. Uživatelské rozhraní musí být vždy funkční a jedině tehdy může být i pěkné.

- **Následnost**

Ovládací prvky mají být rozmístěny podle logiky problému, důležité ovládací prvky podle zásad pro tok ovládacích prvků (shora-dolů, zleva doprava). Následnost může být také podpořena barvami (oko preferuje světlejší objekty před tmavšími, barevné před černobílými, syté barvy před jemnými), seskupením (oko preferuje oddělené ovládací prvky před seskupenými prvky), prezentací (oko preferuje symbol před textem), rozměry (oko preferuje velké objekty před malými) nebo tvarem (oko preferuje jednoduché a obvyklé tvary).

- **Jednotnost**

Ovládací prvky v dialogu by měly tvořit jeden vizuální celek. Pro ovládací prvky stejného významu používáme tytéž rozměry, tvary, barvy a podobně.

- **Proporce**

Při rozmisťování ovládacích prvků klademe důraz na celek, na soulad jednotlivých částí, rozměry a jejich vzájemné poměry. Existuje několik poměrů stran, které jsou obecně považovány za estetické.

- **Seskupování**

Pro zachování přehlednosti je vhodné prvky logicky seskupovat podle významu a účelu. Problematiku rozmisťování ovládacích prvků hlouběji rozpracovává například v pravidlech[Gal97].

Více se lze k této kapitole dočíst na [1].

2.3 Mentální model a nejčastější chyby při tvorbě uživatelského rozhraní

2.3.1 Mentální model

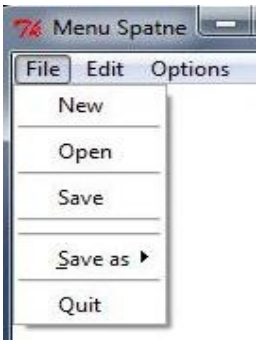
Mentální model představuje vnitřní reprezentaci svého okolí (aplikace), kterou si vytváří ve své hlavě. Jedná se tedy o abstraktní představu vytvořenou uživatelem o tom, jak určitá věc (aplikace) funguje. Při vytváření mentálního modelu je třeba brát zřetel na znalosti, očekávání a zkušenosti uživatele. Cílem je tedy vybudovat takovou aplikaci, kterou uživatel rychle pochopí a rychle se s ní naučí pracovat, popřípadě ovládat. S tímto jsou však spojeny i nejčastější a tolik typické chyby, které se vyskytují už při samotném návrhu dané věci či aplikace.

2.3.2 Typické chyby při tvorbě uživatelského rozhraní

Chyb, které se mohou vyskytovat v uživatelském rozhraní je mnoho. Chyba se může vyskytovat od horního menu přes barevnost, až po celkový obsah aplikace či stránek. Nejčastější typy chyb tedy jsou:

- **Menu**

Je důležité vytvořit přehledné menu, aby se v něm uživatel rychle orientoval a ulehčilo mu to práci. Při tvorbě menu je nutné co nejlépe rozložit primární tlačítka menu a poté zvolit obsah. Tlačítka, které uživatel nevyužije po startu aplikace, je dobré deaktivovat. Neméně důležité je pro celkový přehled v menu i oddělování tlačítek tzv. separátor viz. obrázek 2.9. Na obrázku 2.10 je vidět menu které není přehledné. Za vše mohou nadměrně použité oddělovače.

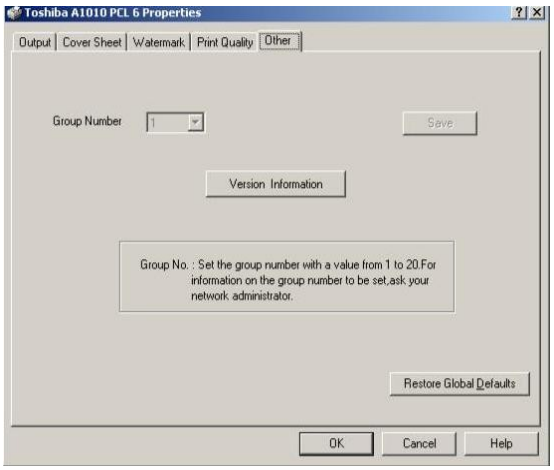


Obrázek 2.9: Správné menu *Obrázek 2.10: Špatné menu*

- **Obsah aplikace či stránek**

Přehlednost a srozumitelnost k uživateli. Tři rámečky s rozděleným obsahem zajišťují celkový přehled viz. Obrázek 2.11. Oproti tomu na obrázku 2.12 je znázornění nerovnoměrně aplikovaných komponent. Tlačítka rozmístěna všude po okně aplikace, což je nepřehledné. Obrázky 2.11 a 2.12 jsou použity z [2].

Obrázek 2.11: Obsah správně [2].

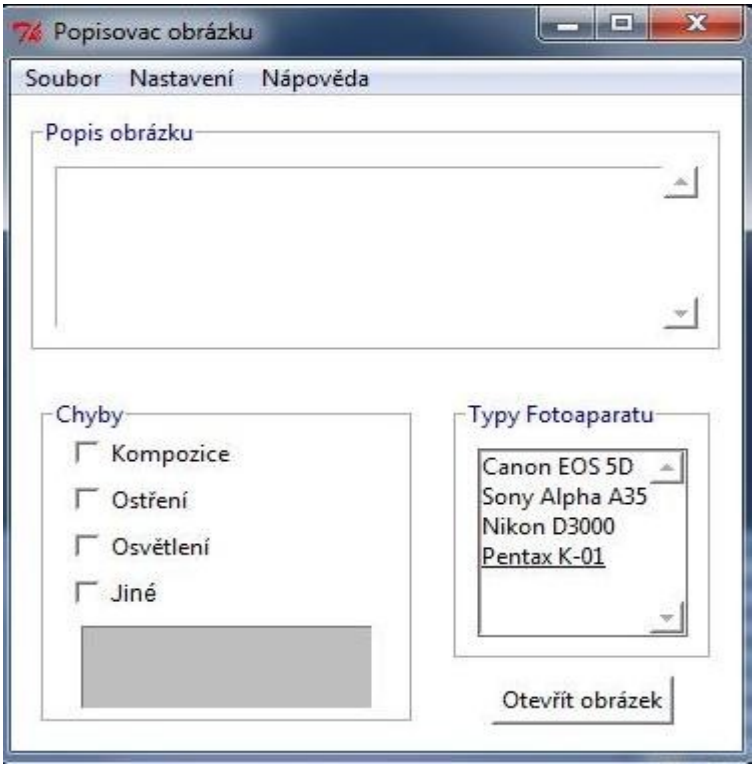


Obrázek 2.12: Obsah špatně [2].

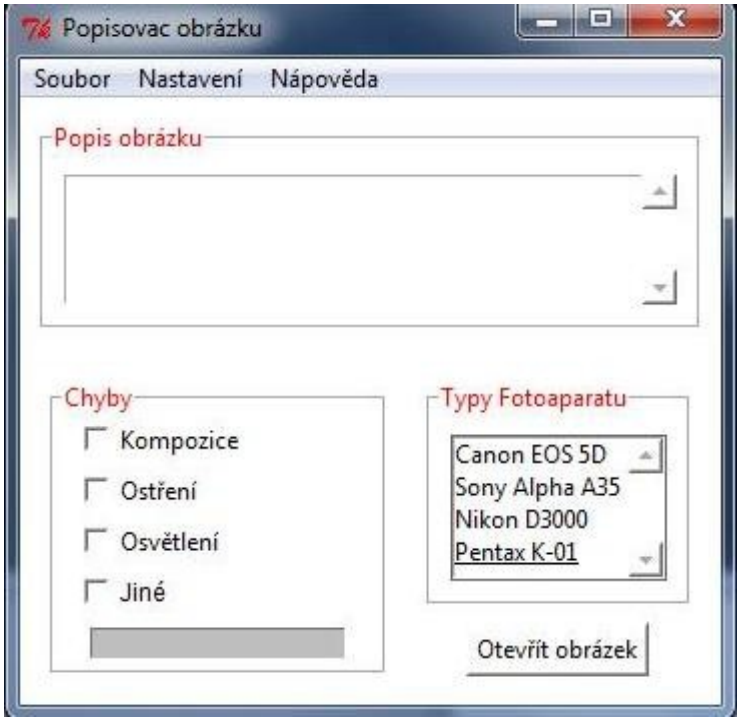
- **Barevnost a nadměrné grafické prvky**

Volba dobré kombinace barev je taktéž důležitá pro tvorbu dobrého mentálního modelu. Pro aplikaci je dobré volit barvy tak, aby pomohly uživateli k lepší orientaci v dané aplikaci. Barvy v prostředí je dobré i tak používat omezeně. Je lepší udělat nadpis nad nějakým rámečkem klasickým písmem, třeba i tučným, než zbytečné volit barvu. Vyhýbat se barvám jako je červená, ostře žlutá, či jiná velice kontrastní barva je základ pro tvorbu dobrého uživatelského rozhraní. Na obrázku 2.14 je dobře vidět, že červená barva pro zvýraznění nadpisu v rámečku není to pravé. V tomto případě vzbuzuje dojem

nějaké chybové hlášky, než jako nadpisu rámečku v aplikaci. Na obrázku 2.13 je barevná kombinace dobrá.

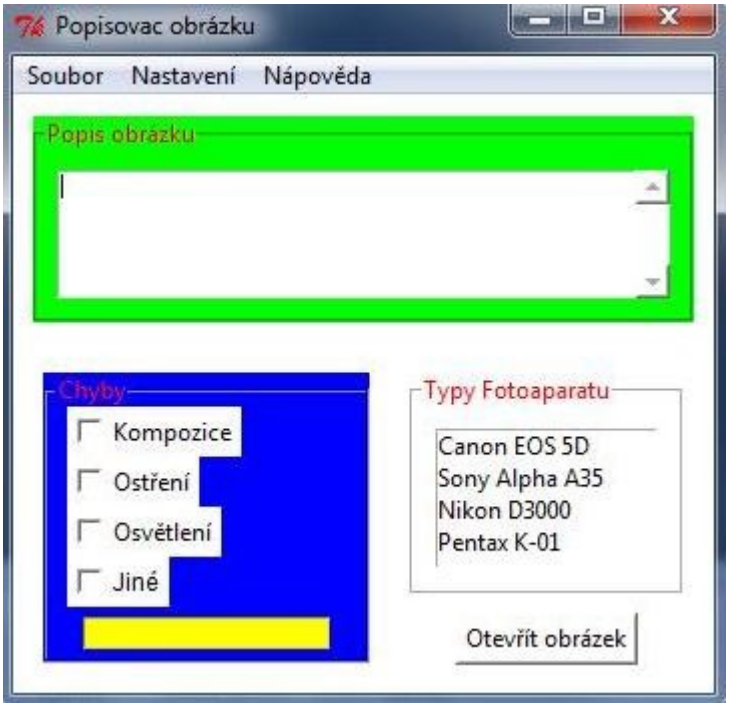


Obrázek 2.13: Barevnost správně



Obrázek 2.14: Barevnost špatně

K chybám v GUI také patří nadměrné grafické prvky v aplikaci či stánkách. Důležité je vyhnout se různým animacím, špatným barevným kombinacím jako na obrázku 2.15.



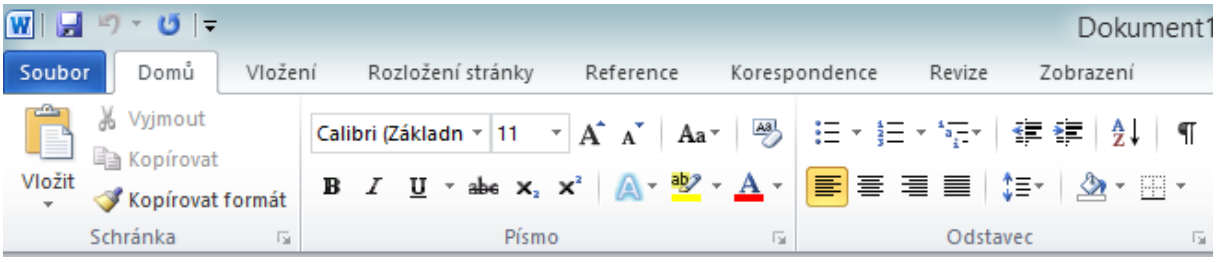
Obrázek 2.15: Nadměrná grafika

3 Zhodnocení uživatelského rozhraní textových editorů

V dnešní době je velká spousta různých textových editorů či procesorů, ale málo kdy setkáme se špatně navrženým uživatelským rozhraním, obzvláště se to týká špičkových licencovaných programů jako je například Word či Wordpad od společnosti Microsoft nebo volně šiřitelný textový editor PSpad. Velmi oblíbeným volně šiřitelným textovým procesorem, který je dostupný pro Linux i Windows je Writer z balíčku LibreOffice. V dalších bodech následuje hodnocení uživatelského rozhraní aplikací.

3.1 Zhodnocení uživatelského rozhraní Microsoft Word

Aplikace Word je součástí balíčku Office od společnosti Microsoft. Jak bylo zmíněno tyto licencované aplikace jsou už na opravdu vysoké úrovni a těžko se zde vyskytuji nějaké chyby z hlediska uživatelského rozhraní. Okno aplikace se dělí na horní menu, ovládací panely, a textový box pro práci s textem. V levé části aplikace se pak vyskytuji nástavbové okna, které jsou napojeny na hlavní, tedy rootovské okno. Menu aplikace je na první pohled řešeno klasickým způsobem, jen místo klasických tlačítek je zde použito záložkové menu. Toto řešení je zvolené pravděpodobně z důvodu umístění podpanelu s nabídkou nástrojů které je umístěno právě pod hlavním menu (viz. Obrázek 3.1). Tyto panely jsou odděleny separátorem neboli oddělovačem. Každý panel má svůj název a obsahuje tlačítka, které odpovídají popisku daného panelu, což je pro celkový přehled v aplikaci velmi důležité. Samozřejmě nechybí ani tlačítka undo a redo, které by měla obsahovat každá aplikace pracující s textem. Každý byť méně zkušený uživatel má možnost se s aplikací rychle seznámit a tedy rychle a účinně s ní i pracovat.



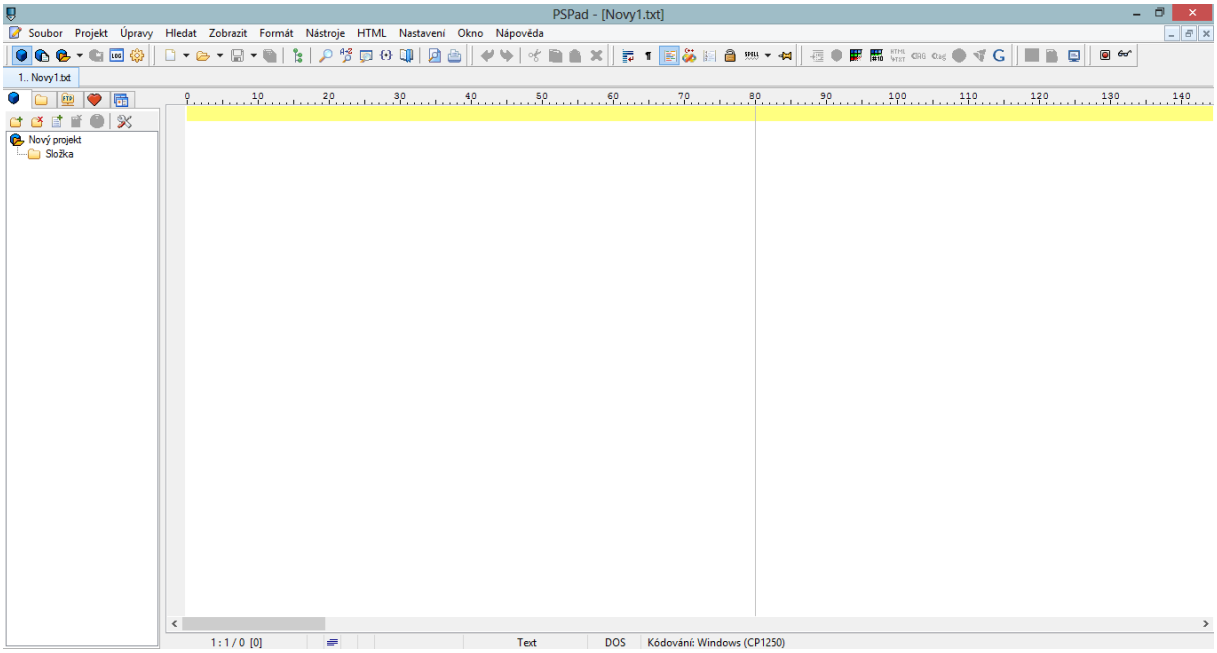
Obrázek 3.1: Menu a panely Microsoft Word

3.2 Zhodnocení uživatelského rozhraní Microsoft Wordpad

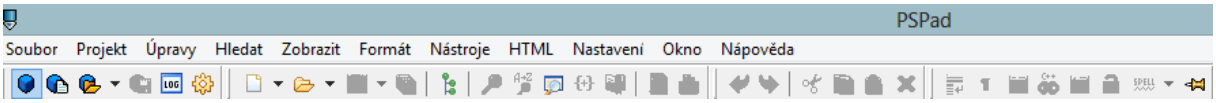
Jelikož Wordpad je produktem od stejného výrobce je nutné předpokládat, že uživatelské rozhraní se moc lišit nebude. V aplikaci Wordpad se vyskytuje stejný návrh uživatelského rozhraní. Jediný rozdíl je v počtu tlačítek. Wordpad je textový procesor, který se pohybuje někde mezi poznámkovým blokem a aplikaci Microsoft Word.

3.3 Zhodnocení uživatelského rozhraní aplikace PSpad

PSpad je velmi oblíbeným univerzálním textovým editorem hlavně pro vývojáře webových stránek a programátory. Vezmeme-li do úvahy, že je jen volně šiřitelným textovým editorem, z hlediska uživatelského rozhraní, je také na vysoké úrovni. Okno aplikace se dělí na panel s klasickým menu a podmenu. Pod menu je umístěna lišta s nejčastěji používanými nástroji. Vlevo je pak zobrazen nástrojový panel. Přes celé okno, jak je tomu zvykem u textových editorů, je umístěn Text box, nad kterým je umístěno jednoduché pravítko. Pro uživatele je určitě velkou výhodou podbarvení aktuálního řádku. Mezi nástrojovým panelem a text boxem je umístěna lišta na otevřené soubory. V uživatelském rozhraní této aplikace je v pořádku, jedinou vadou je snad až příliš mnoho tlačítek v nástrojovém panelu aplikace. Takové množství může zmást nejednoho méně zkušeného uživatele. Vše je ale ovlivněno velkou funkčností editoru. Plusem je, že je dbáno na deaktivaci tlačítek, které v daný moment nelze používat (viz. Obrázek 3.3). V menu jsou tlačítka dobře uspořádána a oddělena separátorem. U některých tlačítek, zejména u těch nejpoužívanějších je umístěna i ikona. V dolní části aplikace se vyskytuje informační lišta.



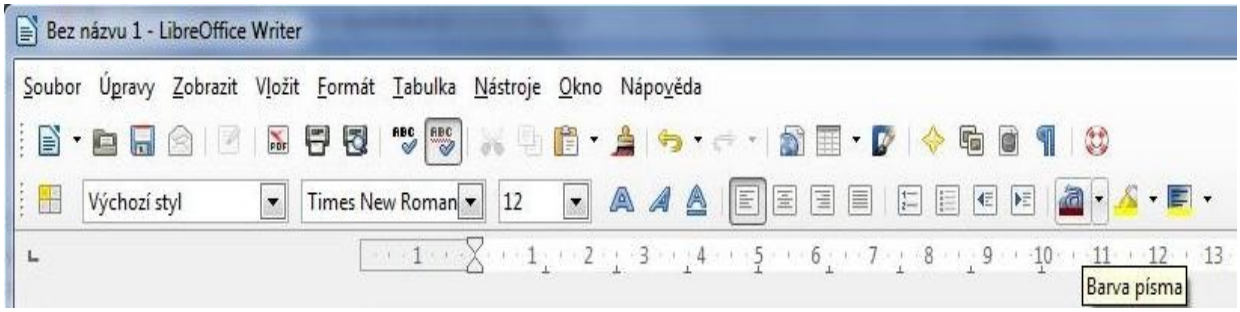
Obrázek 3.2: PSpad



Obrázek 3.3: PSpad deaktivace nepoužitých tlačítek

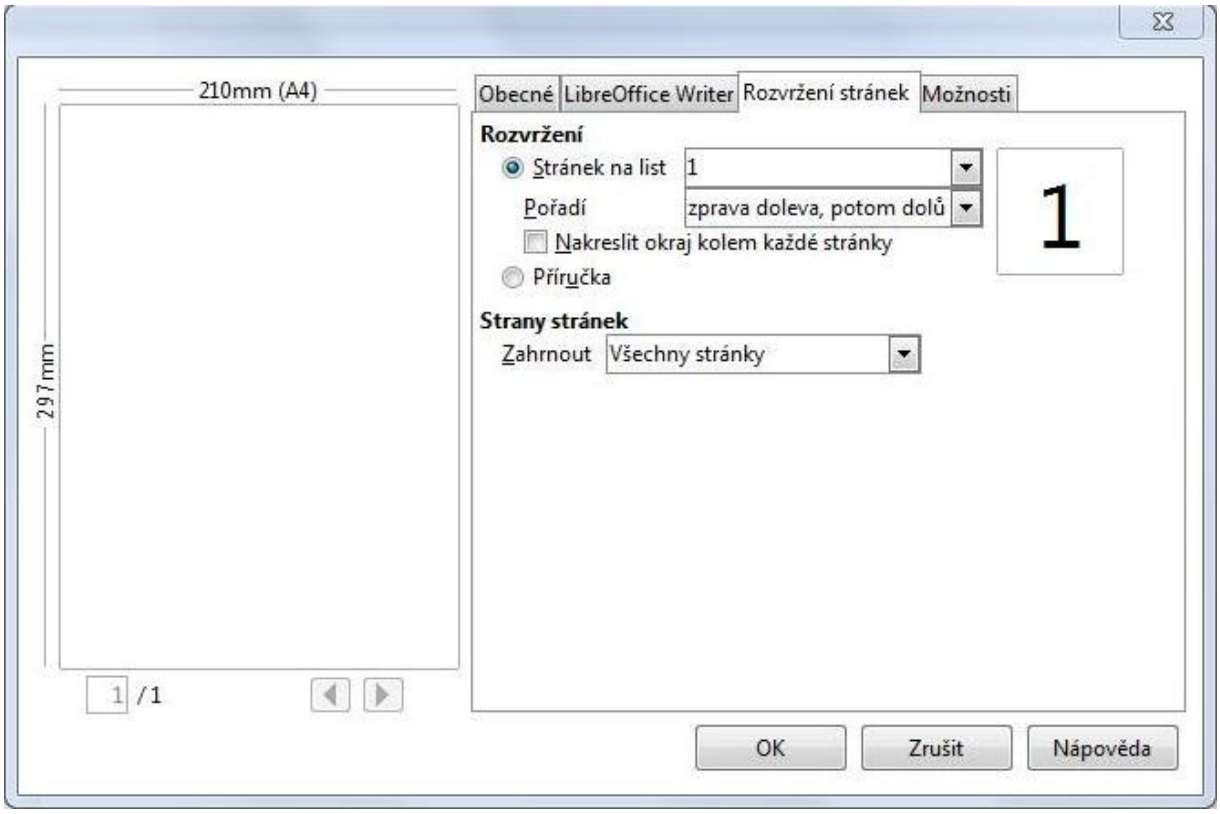
3.4 Zhodnocení uživatelského rozhraní aplikace LibreOfficeWriter

Writer je textový procesor, který je součástí balíčku LibreOffice. Uživatelské rozhraní aplikace je opět navrženo podobným způsobem, jako předchozí aplikace. Okno aplikace obsahuje odshora klasické menu s klasickým navržením podmenu, dva nástrojové panely pro práci s textem. V tomto textovém procesoru se na rozdíl od předchozích vyskytuje více chyb. Jako první a hlavní chybu bych označil nejasné ikony u tlačítek. Běžným pohledem jde jen těžko rozeznat, o které tlačítko se vlastně jedná. Příkladem je tlačítko pro výběr barvy či tučného písma, kurzívy a podtrženého textu, kde jsou pro všechny formátovací tlačítka použity podobné typy tlačítek s písmenem „A“ (viz. Obrázek 3.4). Toto řešení je velmi nezvyklé.



Obrázek 3.4: LibreOffice Writer menu

Dále se zde vyskytuji i chyby rozmístění prvků, rozvržení je dosti chaotické a nepřehledné. Není tedy poznat, co se kde nastavuje (viz. Obrázek 3.5). Je třeba ale vyzdvihnout dobře navržené hlavní menu, které působí přehledně a nechybí v něm deaktivace tlačítek, které v daný moment nejsou v aplikaci potřebné.



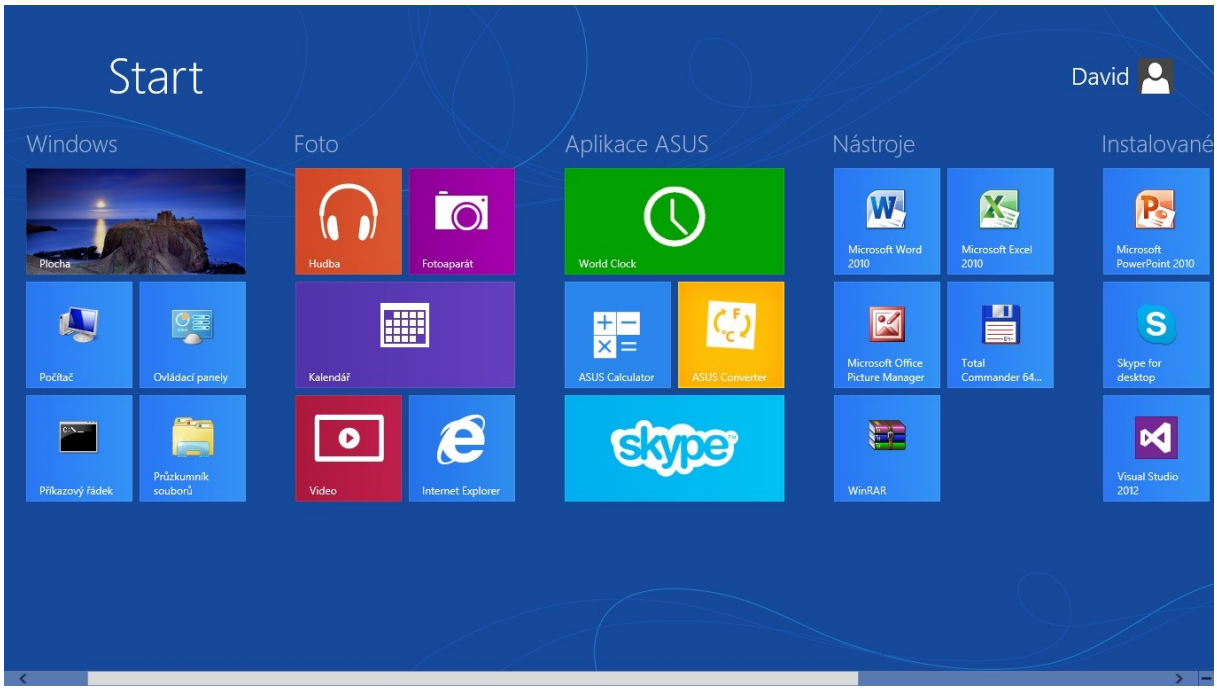
Obrázek 3.5: Příklad špatného rozvržení tlačítek LibreOffice Writer

4 Úvod do prostředí Windows Store

Windows Store je digitální distribuční platformou pro aplikace založené na WinRT, které se obecně nazývají Windows Store aplikace. Windows Store je součástí operačního systému Windows 8 od společnosti Microsoft. Nejnovější Windows podporuje platformy x86, x86_64 a ARM.

4.1 Rozdíly uživatelského rozhraní Windows 8 oproti Windows 7

Windows 8 společnost Microsoft vydala v srpnu roku 2012 jako první Windows určený a připravený pro přístroje s dotykovými displeji. Rozdílů v uživatelském rozhraní obou operačních systémů je tedy opravdu hodně. Hlavním rozdílem z hlediska uživatelského rozhraní je, rozhodně výměna klasického Start panelu, který se objevoval v klasických Windowsech v prostředí Windows Store. Změny vzhledu doznaly jak lišty kolem oken tak i hlavní lišta. Z prostředí Windows 8 zmizela takzvaná „Aero“ transparentní okna. Přesto hlavním a zásadním rozdílem je tedy náhrada tlačítka Start za Windows Store prostředí. Když se poprvé spustí Windows 8 na první pohled to vypadá, že se toho moc nezměnilo, ale opak je pravdou. V každém rohu obrazovky nalezneme něco jiného. Prostředí Windows Store využívá animace k zobrazení panelů po najetí myši, což je samozřejmě zapříčiněno návrhem uživatelského rozhraní pro dotykové displeje. Po najetí do dolního levého rohu, kde se obyčejně vyskytovalo tlačítko Start, zde vyskočí jen malá obrazovka s prostředím Windows Store, které nahrazuje právě tlačítko Start (viz. Obrázek 4.1).



Obrázek 4.1: Prostředí Windows Store

I když dotykem by se prostředí Windows Store ovládalo pohodlně, tak se dá vše dobře zvládnout i s klasickou myší. Návrh samotného prostředí Windows Store vypadá vzhledově velmi dobře. Pokud se uživateli nelíbí základní barevné schéma, může si zvolit barvu dle vlastního uvážení. Veškerá tlačítka v tomto prostředí jsou realizována v obdélníkovém tvaru. Nabídka je také obohacena o panely, které si může vytvořit sám uživatel a umístit v nich své nejčastěji používané aplikace. Pro detailnější nastavení stačí kliknout na pravé tlačítko myši. Z dolní části obrazovky se pak zobrazí lišta s možnostmi nastavení. Toto prostředí tedy vypadá dobře, ale přehled tlačítka Start ani zdaleka nenahradilo. První problém nastává, pokud neznáte klávesové zkratky, když jde o vypnutí počítače. Toto tlačítko se vždy vyskytovalo v nabídce Start, teď se v ní ale nevyskytuje. Nepomůže ani velmi šikovný systém vyhledávání v prostředí Windows Store, pomocí kterého se dá rychle vyhledat téměř vše. Nastává tedy otázka, kde se v uživatelském rozhraní vyskytuje tlačítko vypnout? Pro tento účel je vymezen zase jiný roh obrazovky. A to jak horní, tak i dolní pravý roh. Po najetí do jakéhokoliv zmíněného rohu se zobrazí panel s nabídkou tlačítek Hledat, Sdílet, Start, Zařízení a Nastavení (viz. Obrázek 4.2).

Obrázek 4.2: Lišta vpravo

Návrh umístění celkem důležitého tlačítka pro vypnutí přístroje považuji za jednu z největších chyb z hlediska uživatelského rozhraní nového prostředí Windows 8. Tlačítko se totiž vyskytuje v nabídce nastavení, spolu s nastavením jasu, zvuku a jim podobných tlačítek (viz. Obrázek 4.3). Pro běžného uživatele je tedy otázka jak vypnout přístroj časově náročnější. V levém horním rohu jsou pak k dispozici neuzavřené a používané aplikace.

Obrázek 4.3: Tlačítko Vypnout v Nastavení

4.2 Základy pro vývoj aplikací Windows Store

Aplikace pro Windows Store představují nový typ aplikací, které se prodávají přes web Windows Store a používají na zařízeních s Windows 8. Umožňují snadnou instalaci a čistou odinstalaci. Běží v jednom okně, které ve výchozím nastavení zaplňuje celou obrazovku. Automaticky využívají různé možnosti vstupu, jako jsou dotyková obrazovka, pero, myš a klávesnice. Namísto statických ikon používají živé dlaždice, které mohou zobrazovat různé typy oznámení. Aplikace pro Windows Store je možné psát v několika jazycích, například v C# a Visual Basic s XAML, C++ s XAML nebo DirectX, JavaScript s HTML/CSS. Základem jak začít vyvíjet Windows Store aplikace je stažení jednoho ze dvou bezplatných nástrojů od společnosti Microsoft. Jsou jimi Microsoft Visual Studio Expres 2012 a Blend pro Visual Studio. Oba tyto nástroje spolu vzájemně spolupracují a umožnují vyvíjet a testovat aplikace pro Windows Store prostředí. Ve druhém bodě je pak potřeba po spuštění Visual Studia na žádost aplikace stáhnout vývojářskou licenci. Vývojářskou licenci je možné také získat na oficiálním webu Microsoftu. Vývojářská licence pro nekomerční využití je také zdarma. Další základní informace je možné si přečíst zde [3].

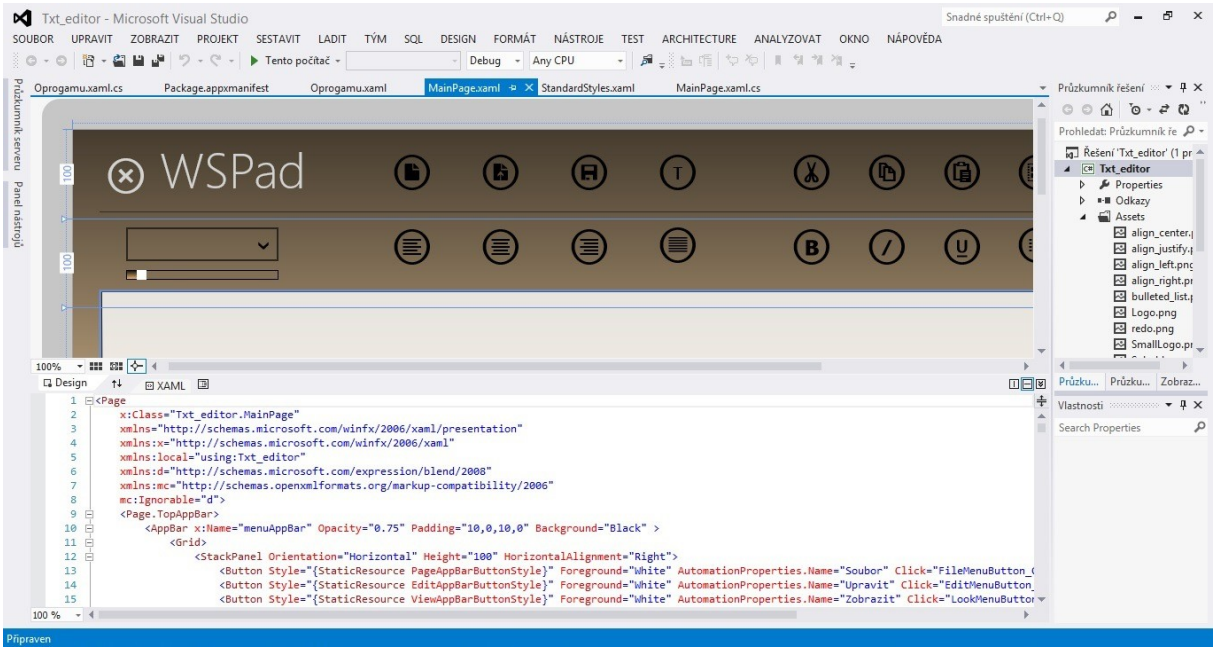
4.2.1 Plánování a návrh

Z hlediska plánování aplikace pro Windows Store nejde ani tak o funkčnost aplikace, ale hlavně o to v čem by aplikace měla vynikat. Využití pravidel pro uživatelská rozhraní a principu návrhu může usnadnit návrh aplikace, která by pak měla být atraktivní a snadno použitelná. Pravidla pro uživatelské prostředí vysvětlují, jak navrhovat aplikace, které poskytují konzistentní, elegantní a poutavé prostředí, a nabízejí rady pro práci se specifickými prvky návrhu (například rozložení) a jednotlivými ovládacími prvky (například seznam). Principy návrhu jsou jakési plány, podle kterých můžete postupovat při návrhu běžných funkcí uživatelského rozhraní, jako jsou ovládání, příkazy, dotyková interakce, reklama a propagace. Další informace lze zjistit zde [3]. Obrázek 4.4 je použit z [4].

Obrázek 4.4: Plánování a návrh [4]

4.2.2 Vývoj Windows Store aplikací pomocí designeru

Při vytváření aplikace pro Windows Store prostředí je k dispozici snadno použitelné API rozhraní, které zjednodušuje prezentaci a rozložení aplikace. Uživatelské rozhraní je možné navrhnou pomocí velmi šikovného návrháře, či designeru, který je dostupný ve Visual Studiu 2012 od společnosti Microsoft. Nelze však spoléhat pouze na designera, vzhled aplikace je možné doladit i v samotném kódu. Designer obsahuje celou řadu komponent od tlačítek až po různé typy Text box. Samotný vývoj aplikace pro prostředí Windows Store je tedy uživatelsky přívětivý. Funkční část pak zajišťuje jeden z dostupných jazyků pro Windows Store aplikace, které jsou uvedeny v bodu 4.2. Dochází tedy k dodržení jednoho ze základních pravidel. Funkční a vzhledová část aplikace je oddělena v kódu. Vzhledová část je psána v jazyku Xaml o funkčnost aplikace se pak programují v jazyku C#, Visual Basic, C++. Pro vývoj webových stránek pak slouží jazyk JavaScript z hlediska funkčního v kombinaci s CSS stylem. Samotná práce s designerem je pro vývojáře aplikací velmi pohodlná. Po výběru komponenty z panelu nástrojů stačí pouze přetáhnout danou komponentu na plátno s hlavním návrhem aplikace. S danou komponentou pak lze volně pohybovat dle vlastního uvážení. Barva a jiné nastavení komponenty se pak dají upravit buď v kódu, nebo přímo v nabídkovém panelu. Na obrázku 4.5 je tento panel vidět vpravo dole.



Obrázek 4.5: Visual Studio 2012 designer

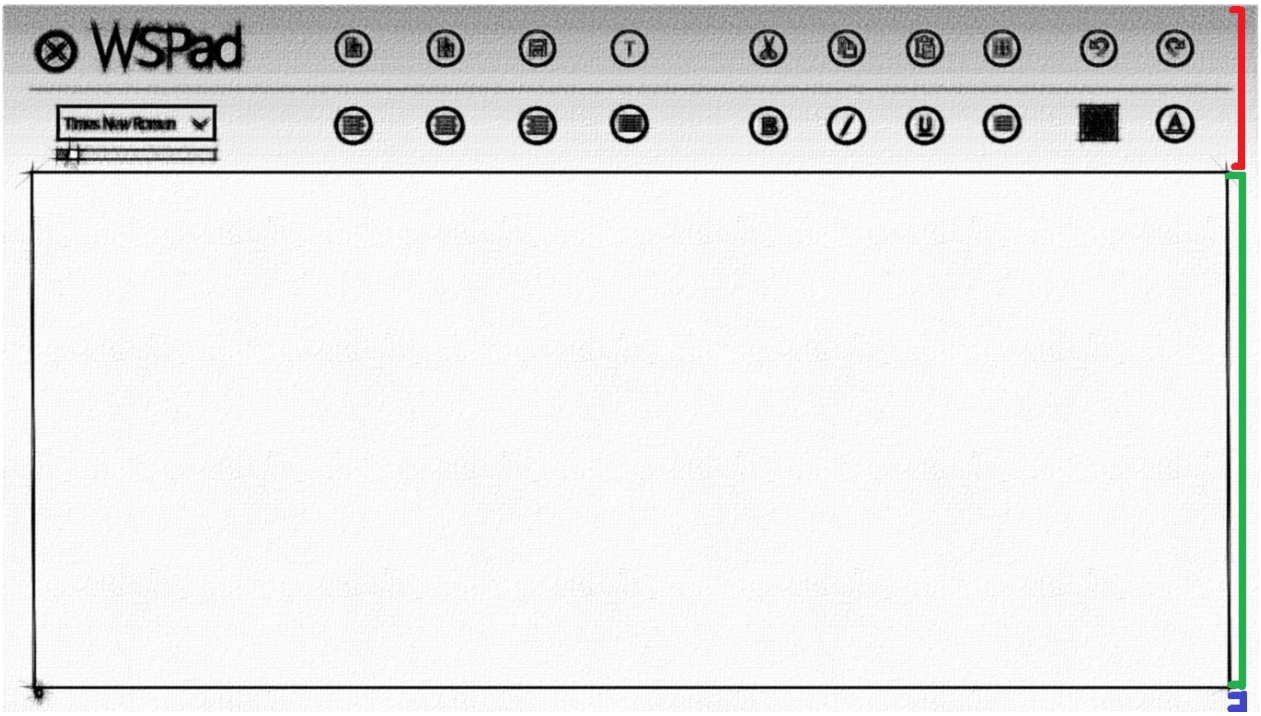
4.3 Návrh a realizace uživatelského rozhraní aplikace pro prostředí Windows Store

Pří vývoji aplikace pro prostředí Windows Store jsem si vybral oblast textových editorů či procesorů. Při výběru této oblasti jsem se rozhodl z několika důvodů. Prvním z nich je zájem o návrh uživatelského rozhraní pro textové editory. Druhým důvodem je, že pro Windows Store prostředí nebyl doposud navrhnut oficiální textový editor s novým uživatelským rozhraním. Realizoval jsem tedy textový editor, který nese název WSPad.

4.3.1 Základní plánování návrhu uživatelského rozhraní aplikace WSPad

Během samotného plánování návrhu uživatelského rozhraní pro výše zmíněný textový editor bylo nutné zaměřit se na přehlednost a použitelnost aplikace, která bude určena pro základní práci s textem v prostředí Windows Store. Aplikace se dělí na tři panely. První panel (na obrázku 4.6 červeně) je ovládací panel pro základní ovládací tlačítka editoru. Druhý panel je určen pro práci s textem. Poslední je pak informační lišta. Samotný první panel je rozdělen malou lištou, která děli tlačítka pro práci s textem a tlačítka pro práci se samotným souborem. Rozměry mezi tlačítky jsou přesně dány společnosti Microsoft, nemělo by tedy docházet k vzájemnému překrývání tlačítek. Ve druhém panelu (na obrázku 4.6 zeleně) tedy textovém je umístěn textový box pro práci s textem. Jelikož je návrh určen pro textový editor, pak tento panel musí logicky velikostně převažovat nad ostatními panely. Druhou variantou by pak bylo roztáhnout textový panel přes celou obrazovku a veškeré ovládací prvky vložit do skrytých lišt. Zvolil jsem však řešení s ovládacím panelem, textovým boxem, tak aby uživatel mohl případně i s nějakým dotykovým přístrojem aplikaci pohodlně používat.

Posledním panelem (na obrázku 4.6 modře) je již zmíněná informační lišta, která obsahuje informace o souboru, počtu znaků v textu. Celkový návrh základního grafického rozhraní aplikace je možné shlédnout na obrázku 4.6.



Obrázek 4.6: Černobílý grafický náčrt rozhraní aplikace

4.3.2 Návrh uživatelského rozhraní aplikace WSPad

Po naplánování vzhledu aplikace je nutné přistoupit k realizaci. K programování aplikace jsem využil služeb designeru Visual Studia 2012. Více informací o samotném designeru naleznete v podkapitole 4.2.2. Samotný takzvaný předběžný naplánovaný návrh nám pouze řekne, jak by asi měla aplikace vypadat. Vzhled aplikace je řešen v jazyku XAML, funkční oblast jsem pak programoval v jazyku C#. V následujících podkapitolách je popsána tvorba aplikace WSPad, od spuštění projektu, přes návrh uživatelského rozhraní až po řešení nutné základní funkčnosti aplikace.

4.3.3 Založení projektu Windows Store aplikace

Po úspěšném spuštění Visual Studia 2012 si lze zvolit nový projekt. První na co je třeba si dát pozor při založení nového projektu základní rozložení aplikace. V nabídce pod záložkou Windows Store jsou k dispozici tři možné varianty základních návrhů. První ze tří možných je rozložení, které nese název Grid App. Jedná se o tří stránkový projekt Windows Store, který umožňuje procházení seskupenými položkami, které jsou vedle sebe uspořádány v mřížce. Podrobnosti jednotlivých položek v mřížce jsou pak zobrazeny na samostatných stranách.

Toto rozložení lze dobře uplatnit při tvorbě webových stránek. Nejznámějším reprezentantem toho rozložení je bezesporu samotná realizace startu ve Windows 8. Druhým možným rozložením aplikace je Split App. Jedná se o dvoustránkový projekt, který také umožňuje procházení seskupenými položkami. První stránka umožňuje výběr skupiny a druhá zobrazuje seznam položek spolu s podrobnostmi pro vybranou položku. Toto rozložení nalezne uplatnění pro nějakou zpravodajskou aplikaci, jako třeba internetové noviny. Posledním z možných rozložení je Basic Page, neboli v nedoslovném českém překladu prázdná stránka. Toto rozložení jsem sám aplikoval pro aplikaci WSPad. Jedná se o jednostránkový projekt pro prostředí Windows Store, ve kterém, jak už napovídá název, nejsou předdefinovány ovládací ani jiné prvky pro rozmístění komponent. Tento návrh rozmístění ocení především ten vývojář, který chce vytvořit aplikaci podle svého vlastního návrhu. Po zvolení rozložení a dalších základních úkonech, jako je pojmenování projektu a jiné, se pak vývojáři zobrazí designer a hlavní pracovní plocha. Vývojové prostředí nám samo vygeneruje soubory App.xaml, MainPage.xaml, ke kterým se vážou soubory v jazyku C# se stejným názvem. Je tedy postaráno o rozdělení kódu z hlediska uživatelského rozhraní a funkčnosti aplikace, jak už je zmíněno v podkapitole 4.2.2. V mém případě jsem použil jen některé základní soubory pro tvorbu Windows Store aplikace. Informace o souborech obsahuje následující popis:

1. **MainPage.xaml**
   Obsahuje základní vzhled aplikace, k souboru lze přistoupit i pomocí designeru. Váže na sebe stejnojmenný soubor v jazyku C#, který zaručuje funkčnost aplikace.
2. **App.xaml**
   Základní soubor, bez kterého nelze spustit aplikaci. Dá se využít k nastavení barev pro celou aplikaci.
3. **StandardStyles.xaml**
   Velmi důležitý soubor. Obsahuje grafické prvky přímo od společnosti Microsoft, které může vývojář využít při vývoji své aplikace.
4. **Package.appxmanifest**
   Soubor, který obsahuje veškeré informace o aplikaci od názvu po až ikonu aplikace.

4.3.4 Tvorba uživatelského rozhraní aplikace WSPad

Po vygenerování souboru MainPage.xaml obsahuje zdrojový kód pouze základní rysy aplikace. Základní tag je tedy `<Page>`, ve kterém začínají, i končí veškeré vzhledové úpravy. V aplikace WSPad jsem zvolil rozložení Grid z důvodu použitelnosti při rozmístění ostatních prvků. Základem vzhledu každé aplikace je volba vhodných barev. Základním barevným prvkem v aplikaci je kombinace barev tan (hnědé)a černé. Důvodem volby těchto barev je dobrá čitelnost a vhodnost zkombinovat právě tyto barvy. Pozadí je řešeno následujícím kódem 4.1.

```
<Grid.Background>
<LinearGradientBrush EndPoint="0.5,0" StartPoint="0.5,0.5">
<GradientStop Color="Black" Offset="1.5" />
<GradientStop Color="Tan"/>
</LinearGradientBrush>
</Grid.Background>
```

Zdrojový kód 4.1: Definování barevného pozadí aplikace

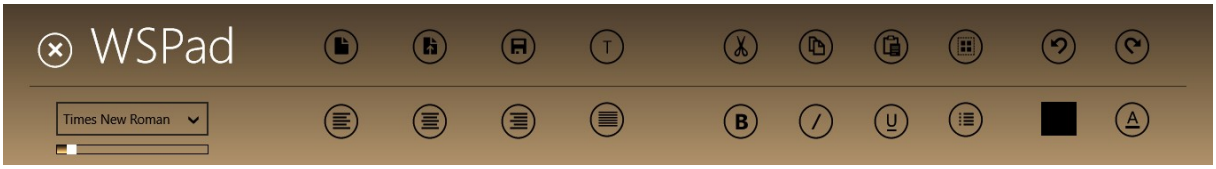
Pozadí je tedy napojeno na Grid a pomocí tagu <LinearGradientBrush> dochází ke kombinaci těchto dvou barev, kde se lineárně vykreslí pro zvolenou oblast. K tomu slouží nastavení vlastností EndPoint a StartPoint, které umožňují přesně nastavit, v jakém bodě se barvy promíchají.

Následně lze podle nakresleného předběžného návrhu aplikace vkládat komponenty na plátno designeru. Kreslené předběžné schéma je k dispozici v bodě 4.3.1. Podle předběžného návrhu jsem rozdělil aplikaci na tři základní panely. Nástrojový, Textový a Informativní panel. K tomu mi posloužila komponenta, která nese název <StackPanel>, který nám zaručí, že všechny vložené prvky (tlačítka, comboboxy, aj.) uspořádá do jednoho řádku, ať už horizontálně nebo vertikálně. Začal jsem tedy v prvním Nástrojovém panelu, který je určen pro veškeré ovládací prvky aplikace. Panel jsem již od předběžného návrhu chtěl rozdělit na dvě poloviny, tak aby tlačítka, která budou souviset pouze s textovým polem jako jsou tlačítka pro tučné písmo, kurzívu nebo tlačítka pro zarovnávaní textu byla oddělena od ostatních tlačítek pro práci se souborem jako takovým. Mezi tyto tlačítka jsem zařadil nahrání nového nebo uloženého souboru, ukládání, a tlačítko pro tisk. Na levé straně panelu je pak umístěn název programu a tlačítko na ukončení celé aplikace. Vzhledem k tomu, že se každý uživatel při práci s aplikací orientuje v programu zleva doprava, je umístění ukončovacího tlačítka a názvu programu celkem na místě. Vytvoření tlačítka v aplikaci je možné řešit dvěma způsoby. Buď přímo v kódu MainPage.xaml nebo pomocí designera. U tvorby jiného klasického uživatelského rozhraní aplikace psané v jiném jazyku například pro Javu bych preferoval metodu první a tedy psaní přímo do kódu. Ve Visual Studiu 2012 při tvorbě aplikace pro prostředí Windows Store je ale o dost lepší využít služeb toho to velmi šikovného designera. Stačí jen najít potřebnou komponentu v nástrojové liště, v tomto případě tlačítko neboli <Button> a pohybem myší jej přetáhnout na plátno designera aplikace. Tlačítko jsem tedy volně vložil do StackPanelu. Ve zdrojovém kódu se poté vygeneruje kód pro dané tlačítko. Syntaxe tlačítka je vidět na následující ukázce zdrojového kódu 4.2.

```
<Button x:Name="btnExit" Click="btnExit_Click" Foreground="LightGray"
Style="{StaticResource NoAppBarButtonStyle}" Margin="10,20,0,0" Width="87"
Height="66" RenderTransformOrigin="0.525,0.969"/>
```

Zdrojový kód 4.2: Ukázka řešení tlačítka

Základem u každé nově vložené komponenty je dobré a přehledné pojmenování. Toto tlačítko pro ukončení aplikace je pojmenováno btnExit. Následující vlastnost Click slouží pro funkční otázku tlačítka. Vlastnost Style je velmi důležitá pro každé tlačítko či prvek.



Obrázek 4.7: Nástrojový panel

Pomocí příkazu `{StaticResource NoAppBarButtonStyle}` řekneme, že chceme, aby dané tlačítko či komponenta dostala předem definovaný vzhled tlačítek od společností Microsoft pro prostředí Windows Store. Tyto tlačítka jsou specifikovány svým kulatým vhledem. Jejich součástí jsou také ikony uvnitř kruhového ohraničení. Aby tato speciální tlačítka byly dostupně k použití, je nutné je odkomentovat ve zdrojovém kódu v souboru StandardStyles.xaml. Řešení ostatních tlačítek je řešeno podobným způsobem. Vzdálenost mezi tlačítky je přesně definována společností Microsoft, nelze tedy záměrně dojít k situaci, kdy se v aplikaci překrývají dvě a více tlačítek nebo jiných komponent. Volba barev u všech komponent v nástrojové liště je černá, krom textu s názvem aplikace a tlačítkem na vypnutí, kde jsem zvolil stříbrné provedení. Na všechny prvky je aplikován příkaz `ToolTipService.ToolTip`, který zaručuje popisek každého tlačítka v aplikaci. Již zmíněné oddělení tlačítek pro práci s textem a souborem jsem realizoval jednoduchou lištou, kterou jsem umístil mezi tyto tlačítka. Nástrojový panel lze prohlédnout na obrázku 4.7.

Druhý panel prostorově nejobsáhlejší je Textový panel. Textový panel obsahuje pouze textové pole pro práci s textem. U textových editorů nebo procesorů je běžnou věcí, že textové pole je velikostně nejobsáhlejší komponentou v celé aplikaci. Pří návrhu uživatelského rozhraní jsem poprvé použil klasicky TextBox, který slouží pro práci s textem na základní úrovni. Při zjištění více informaci leze zjistit, že pro Windows Store aplikace je vytvořena staronová komponenta, která nese název RichEditBox. Jedná se o takovou upravenou komponentu, která se používá při tvorbě aplikací pro Windows Form, kde nese název RichTextBox. I když název je téměř totožný, použitelnost a metody při práci jsou ale dosti rozdílné. Tento textový box jsem tedy použil a nahradil jsem jím klasicky TextBox. Postupem času se ukázalo, že to byla správná varianta. Jak už z názvu vyplývá, má bohatší vlastnosti (pokročilejší formátování textu, odřádkování) než klasický TextBox a pro textový editor či procesor je použíti RichEditBoxu určitě dobrou variantou. Syntaxi RichEditBoxu je možné vidět na následujícím zdrojovém kódu 4.3, kde je opět pojmenování komponenty, které je velmi důležité pro funkční část, dále pak další typické věci pro textové pole, jako je velikost písma, ohraničení textového pole a další podobné vlastnosti. Co je navíc oproti klasickému textovému poli je našeptávač, který lze aktivovat nebo deaktivovat příkazem `IsSpellCheckEnabled`. Našeptávač jsem deaktivoval,

jelikož neobsahoval české texty. Z hlediska napojení na funkční část pro textové pole slouží příkaz TextChanged, který odkazuje na metodu ve funkční části aplikace.

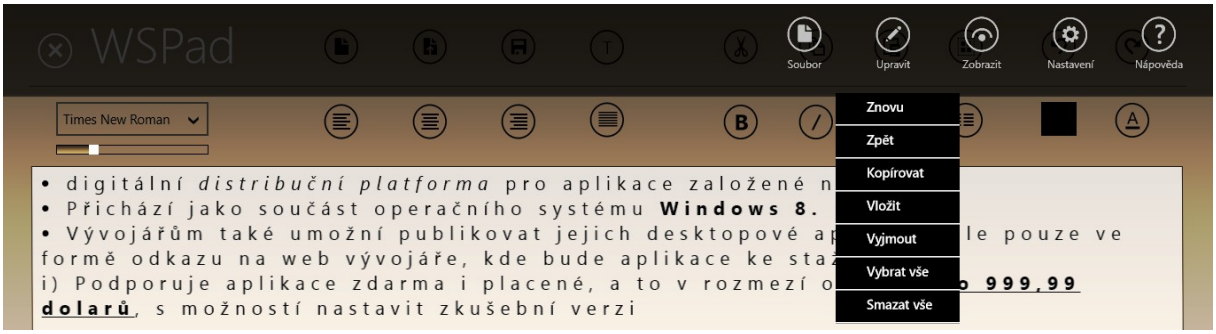
```
<RichEditBox x:Name="txtRichBox" BorderBrush="Black" FontSize="12"
IsSpellCheckEnabled="False" BorderThickness="1" HorizontalAlignment="Left"
Margin="34.16,-18,0,0" VerticalAlignment="Top" Height="560"
Width="1299.991" Grid.Row="2" RenderTransformOrigin="0.5,0.5"
UseLayoutRounding="False" d:LayoutRounding="Auto"
TextChanged="txtRichBox_TextChanged_1">
```

Zdrojový kód 4.3: Řešení Rich Edit Boxu

Posledním panelem, který je umístěn v aplikaci WSPad je Informativní panel. Tento panel je v dolní části aplikace. Uživatel si jej při prvním seznámení s aplikací ani nemusí všimnout. Mimo čísla o počtu znaků v textovém poli totiž ze začátku nezobrazuje žádné další informace. V této liště se mimo čísla na počet znaků vyskytují i informace o nahraném souboru, kde je uveden jak název souboru, tak i čas posledního uložení. Tyto informace o souboru se však pochopitelně zobrazí pouze při načtení nějakého z dostupných souborů.

Základní vzhled byl tedy hotov. Chybělo už jen vyřešit jak udělat v aplikaci WSPad hlavní menu, které lze vidět na obrázku 4.8 a případné vedlejší okna aplikace, jako je například okno pro nastavení a tisk. V prostředí profesionálních aplikací Windows Store od společnosti Microsoft si lze všimnout, že veškeré vedlejší okna nebo hlavní menu, které je u klasických uživatelských rozhraní řešeno v horní liště aplikace je v prostředí Windows Store skryté a zobrazuje se, pouze když jej uživatel potřebuje. Hlavní nabídkové menu jsem tedy pro aplikaci WSPad vyřešil pro Windows Store prostředí standardním způsobem. Pro uspořádání tlačítek do hlavního nabídkového menu jsem použil klasický AppBar se StackPanelem a tagem `<Page.TopAppBar>` jsem zajistil, že se menu bude zobrazovat v horní části obrazovky. Důvodem pro toto rozhodnutí je, že uživatel z klasických uživatelských rozhraní je více zvyklý na menu v horní části obrazovky, než dole, kde se u některých aplikací v prostředí Windows Store vyskytuje. Jednotlivá tlačítka v tomto nabídkovém menu jsou řešeny obdobným způsobem jako ostatní tlačítka v aplikaci. Jediným rozdílem je, že pro tlačítka v menu jsem zvolil i popisek pro každé tlačítko. K popisu jednotlivých tlačítek slouží příkaz `AutomationProperties.Name="Soubor"`, kde v uvozovkách je název daného tlačítka podle toho jak chceme tlačítko pojmenovat. Následně je nutné vyřešit podmenu pro hlavní nabídku. Při tvorbě tohoto podmenu jsem se inspiroval klasickým uživatelským rozhraním. Realizoval jsem jej klasickou aplikaci PopUp, kde jsem umístil panel a do něj pak horizontálně vložil veškeré potřebné tlačítka. Největším problém u řešení těchto PopUp oken v prostředí Windows Store je jeho zobrazení v aplikaci. Hlavní řešení zobrazení těchto vedlejších oken se ve Windows Store řeší animací, kterých je celá řada. V aplikaci WSPad jsem pro zobrazení těchto podmenu zvolil animaci, která zobrazí

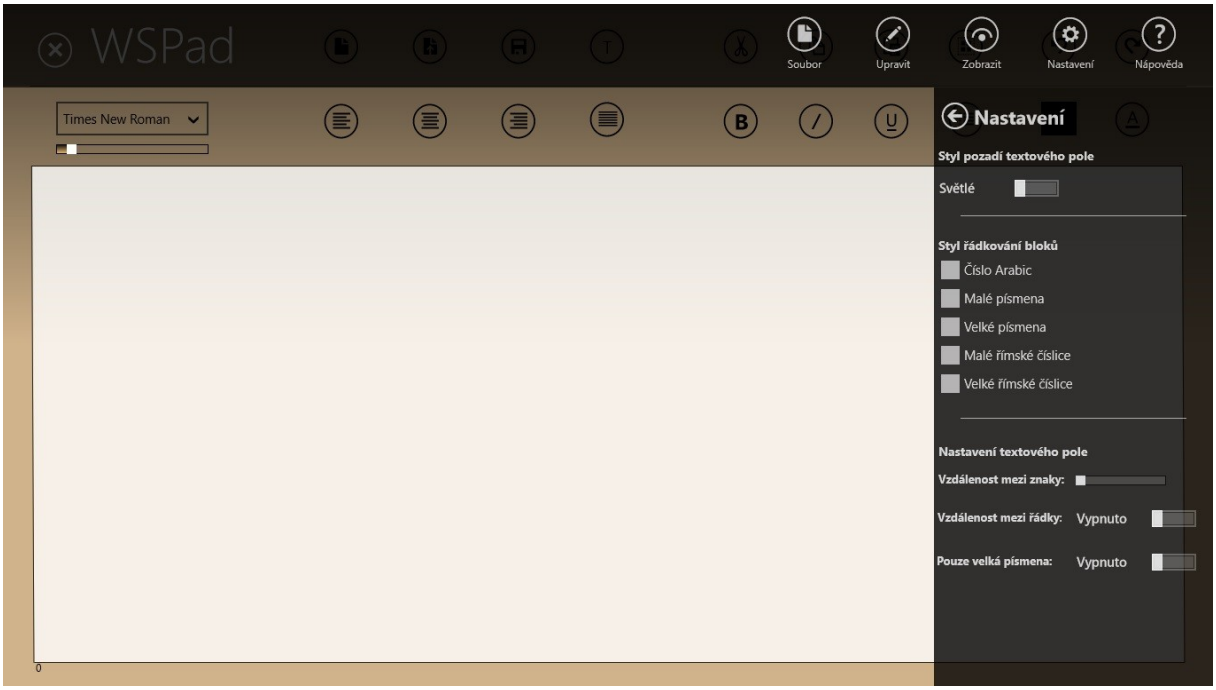
podmenu z pravé strany obrazovky přímo pod dané tlačítko v hlavním menu. Celé menu je barevně provedeno černo barvou. Umístěné komponenty jsou pak v bílém provedení. V podmenu je rozdělení tlačítek řešeno separátorem, vytvořeným pomocí tagu `<Line>`.



Obrázek 4.8: Řešení hlavního menu a podmenu aplikace WSPad

Další co je potřeba vyřešit při tvorbě aplikace pro Windows Store jsou takzvané vedlejší okna aplikace. Pokud se jedná o obsáhlejší aplikace, která vyžaduje více místa doporučuji přepnutí z hlavní rootovské obrazovky na vedlejší. Princip je jednoduchý. Vytvoří se další obrazovka s koncovkou .xaml a ve zdrojovém kódu se připíše přesměrování na danou obrazovku. Toto řešení pro vedlejší okna bych preferoval spíše pro aplikace, které pracují s více obrázky a potřebují tedy i více prostoru. V aplikaci WSPad jsem pro vedlejší okna použil řešení ve stylu prostředí Windows Store s vyskakovacími panely na levém a pravém okraji obrazovky. Vedlejší okna jsem ve své aplikaci řešil dvě. Prvním z vedlejších oken bylo Nastavení (obr. 4.9), druhé pak okno či panel pro Tisk (obr. 4.10). Vyskakovací panel pro Nastavení je řešen pomocí PopUp element, na který je napojen StackPanel a v něm rozmístěny jednotlivé komponenty. Řešení tohoto panelu je tedy dosti podobné hlavnímu nabídkovému menu. Rozdíl je ten, že u nabídkového menu se pomocí tagu AppBar nemusí řešit zobrazení toho panelu na obrazovce. U panelu s nastavením jsem pro zobrazení použil stejně jako v podmenu animaci pro prostředí Windows Store, které se k PopUp elementům napojují v jazyce C#. Animace dodávají aplikaci dotaženost a vylepšují celkový pohled na aplikaci. Panel pro nastavení jsem umístil na pravou stranu obrazovky, protože je blíže k tlačítku ke spuštění tohoto panelu. Panel pro tisk se z hlediska zobrazení moc neliší od panelu nastavení, jen je umístěn na levé straně obrazovky. Důvodem je aby uživatel přesně věděl, kde má jakýkoliv panel očekávat. Při větším počtu těchto panelů bych asi zvolil variantu stejné strany pro veškeré panely. U této aplikace však uživatel po prvním použití bude vědět, že když doje k tisku textu, může náhled očekávat vlevo a při potřebě nastavení nalezne panel zase na pravé straně obrazovky. Také je na něm umístěna jiná animace, která vystihuje nástupu tohoto panelu na obrazovku. Veškeré vyskakovací panely aplikace také disponují návratovou šipkou, která je tolik specifická u aplikací v prostředí Windows Store. Jelikož se jedná o tlačítko opět jsem využil kolekce tlačítek pro toto prostředí. Panel sice lze deaktivovat i pravým

tlačítkem myši, proto z hlediska funkčnosti návratové tlačítko není až tak potřeba, ale dodává celkový vzhled panelu.



Obrázek 4.9: Řešení vedlejších oken aplikace, Nastavení

Obrázek 4.10: Řešení vedlejších oken aplikace, Tisk

4.3.5 Řešení základní funkčnosti aplikace WSPad

Ke každému návrhu uživatelského rozhraní aplikace by měla být i základní funkčnost programu. WSPad disponuje základní funkčností pro práci s textem, nahrávání a ukládaní textových souborů. Z hlediska práce s textem aplikace umí veškeré úpravy písma od tučného písma, přes zarovnání textu až po velikost písma, kterou jsem vyřešil trochu netradičním, ale účelným aplikováním slideru. Pro výběr barvy písma slouží RGB míchač barev. Pokud se uživateli líbí zvolená barva může danou barvu uložit poté i načíst a znovu použít, díky seřazení barev do listu podle uložení. Tento list s namíchanými uloženými barvami lze najít v hlavním menu pod tlačítkem Zobrazit. V aplikaci je k dispozici základních šest typů písma, které lze pohodlně vybrat z nabídky v comboboxu. Veškeré úkony, které mohou ovlivnit práci s textem jsou zabezpečeny MassageDialogem, tak aby nedošlo například ke ztrátě textu při načtení nového souboru. Jakákoliv práce s textem možná díky výskytu RichEditBoxu v aplikaci. Panel Nastavení z hlediska funkčnosti obsahuje různé možnosti úprav pro práci s textem a editorem. Krom klasického bílého pozadí textového pole lze nastavit tmavší provedení, díky speciálnímu tlačítku ToogleSwitch. Jedná je o přepínač dvou možností, neboli „True“ či „False“. Toto tlačítko je velmi povedené a hodí se k aktivaci či deaktivaci jednotlivých funkcí v editoru. Dále lze nastavit styl řádkování bloků pomocí klasických Comboboxu, vzdálenost mezi znaky a řádky, kterou lze nastavit pomocí slideru. Na veškeré grafické prvky při zobrazení jednotlivých panelů a oken je napojena animace, dostupná pro prostředí Windows Store. Příklad je k dispozici na následující ukázce zdrojového kódu 4.4, kde je aplikována animace z kolekce Transition. Jak je zmíněno v bodě 4.3.4 funkčnost aplikace je zajištěna v jazyku C#.

```
panelSettings.Transitions = new
Windows.UI.Xaml.Media.Animation.TransitionCollection();
panelSettings.Transitions.Add(new
Windows.UI.Xaml.Media.Animation.PaneThemeTransition());
```

Zdrojový kód 4.4: Animace Transition

5 Závěr

S uživatelskými rozhraními je možné se setkat téměř všude. Při hodnocení uživatelských rozhraní nejen textových editorů, na které jsem se ve své prací zaměřil ale i u jiných aplikací, které běžně používám, jsem se nesetkal s výraznými chybami.

Hlavním cílem mé bakalářské práce bylo navrhnout a realizovat uživatelské rozhraní aplikace pro prostředí Windows Store. V případě uživatelských rozhraní se jedná ještě stále o novou věc, materiálů a učebních pomůcek mnoho není. Při návrhu aplikace textového procesoru WSPad jsem se mohl setkat s rozdíly v uživatelském rozhraní, které přináší toto nové prostředí. Veškeré aplikace pro toto prostředí jsou realizovány na celou obrazovku bez možnosti minimalizace. Mohou tedy působit dojmem webové prezentace. Výhodou těchto aplikaci je dle mého názoru efektivní a účelné skrytí tlačítek pomocí různých animací, které se však některým uživatelům nemusí zamlouvat.

Největší přínos pro mě spočíval právě v realizaci programu WSPad, protože jsem měl možnost naučit se něco nového. Vědomí, že požadavky uživatelů jsou někdy dosti různorodé a získání schopnosti pohledu na nějakou aplikaci či produkt očima uživatele, mi doufám v budoucnu umožní jednodušeji a efektivněji navrhovat aplikace a jejich uživatelská rozhraní, která možná zpříjemní práci všem uživatelům.

Použitá literatura

[1] DOSTÁL, Martin. Základy tvorby uživatelského rozhraní. [online]. [cit. 2013-03-19]. Dostupné z: http://phoenix.inf.upol.cz/esf/ucebni/gui-dostal.pdf

[2] SOJKA, Eduard. Grafický design GUI Mentální modely. [online]. [cit. 2013-03-19]. Dostupné z: http://mrl.cs.vsb.cz/people/sojka/uro/design1_mental.pdf

[3] *MsdnDev Center* [online]. 2012 [cit. 2013-03-29]. Dostupné z: http://msdn.microsoft.com/cs-cz/windows/apps/jj679957

[4] *MsdnDev Center* [online]. 2012 [cit. 2013-03-29]. Dostupné z: http://msdn.microsoft.com/cs-cz/windows/apps/jj680877

[5] SHNEIDERMAN, Ben a Catherine PLAISANT: *Designing the User Interface: Strategies for Effective Human-Computer Interaction Addison-Wesley*, 2004

[6] DIX Alan, ABOWD Gregory D., BEALE Russell: *Human-Computer Interaction 3rd Ed., Prentice Hall*, 2003

[7] CARROLL John: *HCI Models, Theories, and Frameworks, Morgan Kaufmann*, 2003

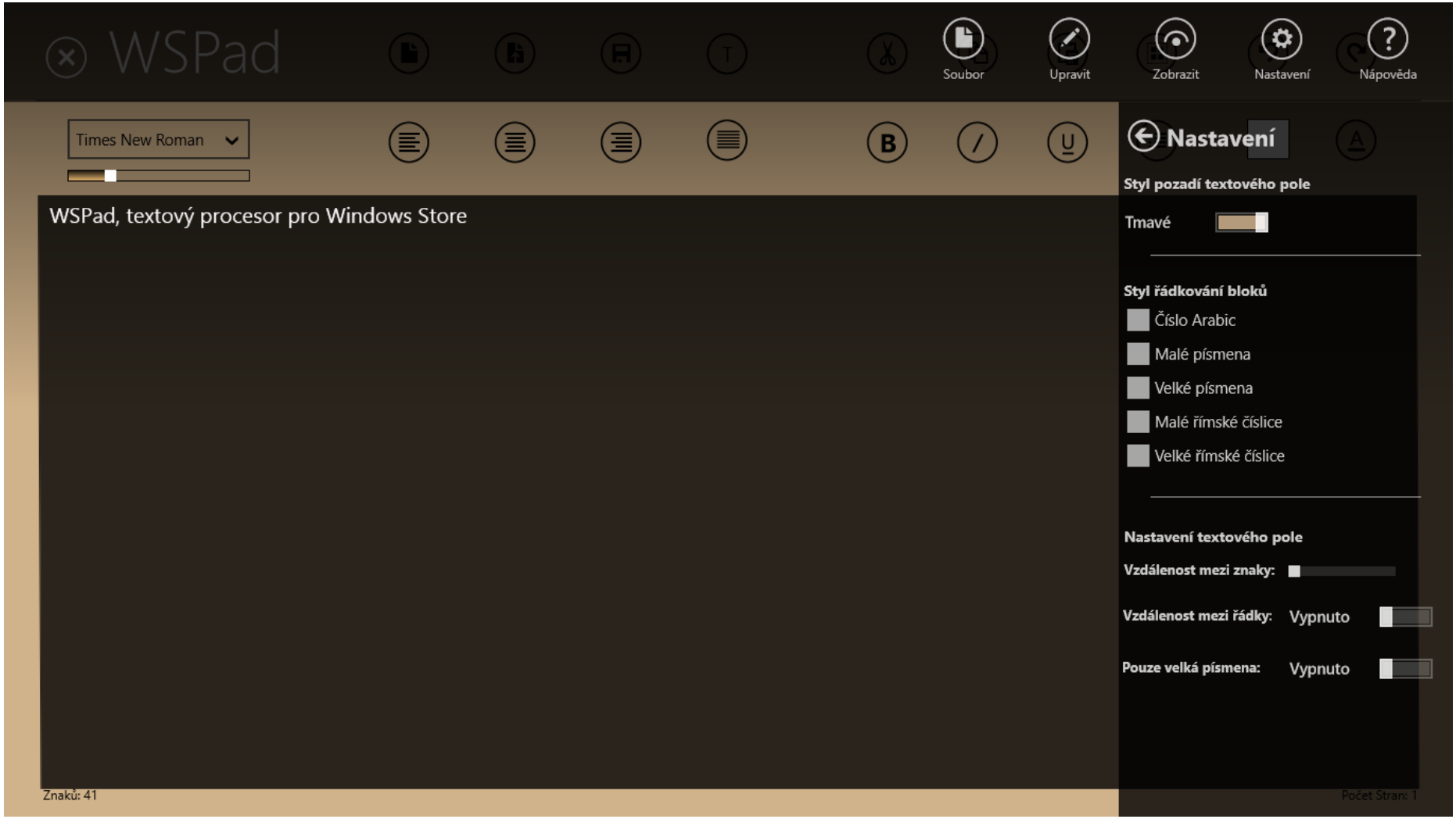
Seznam příloh

Součástí BP je CD/DVD.

Adresářová struktura přiloženého CD/DVD:

- WSPad/
  - Txt_editor.rar
- Textová část/
  - Bc_prace.pdf


WSPad
Times New Roman
B
Znaků: 0
Počet Stran: 1


WSPad
Soubor
Upravit
Zobrazit
Nastavení
Nápověda
Times New Roman
B
WSPad, textový procesor pro Windows Store
Znaků: 41
Nastavení
Styl pozadí textového pole
Tmavé
Styl řádkování bloků
Číslo Arabic
Malé písmena
Velké písmena
Malé římské číslice
Velké římské číslice
Nastavení textového pole
Vzdálenost mezi znaky:
Vzdálenost mezi řádky:
Vypnuto
Pouze velká písmena:
Vypnuto
Počet Stran: 1